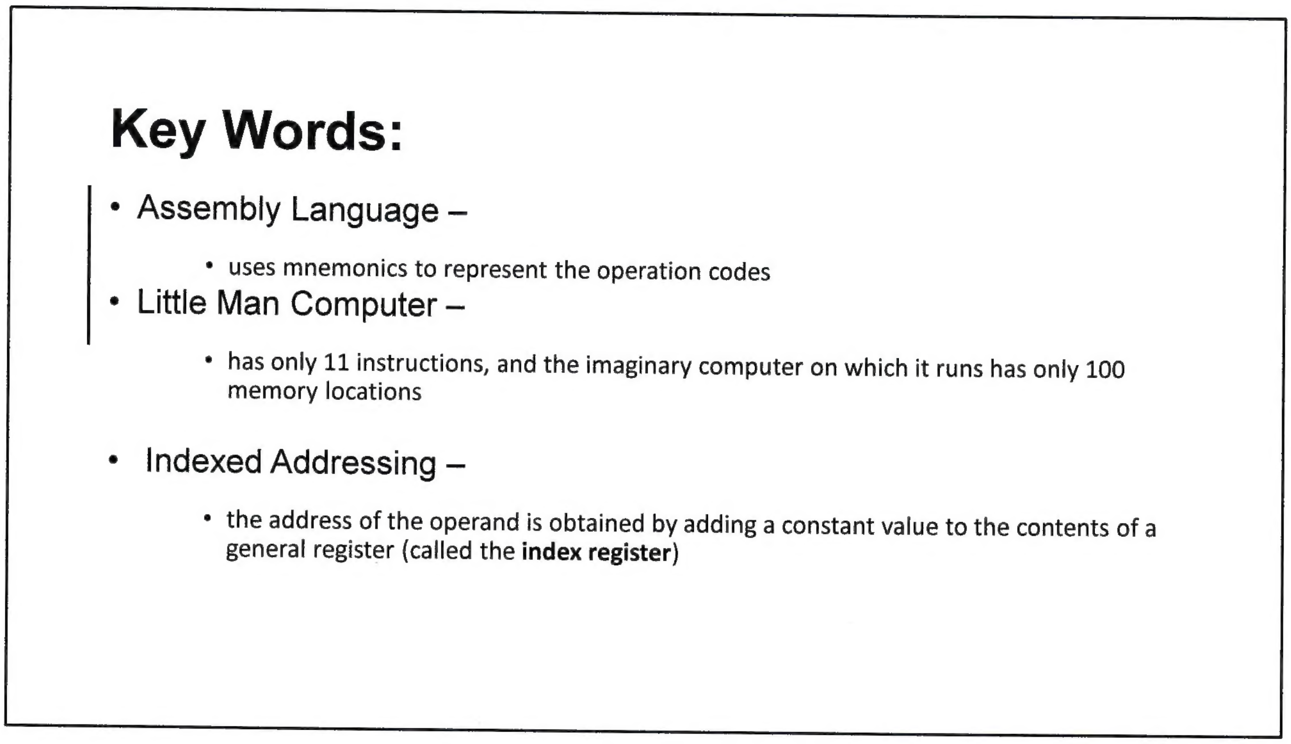
Assembly language

Section 3 – Chapter 14

Monday, 22 January 2024

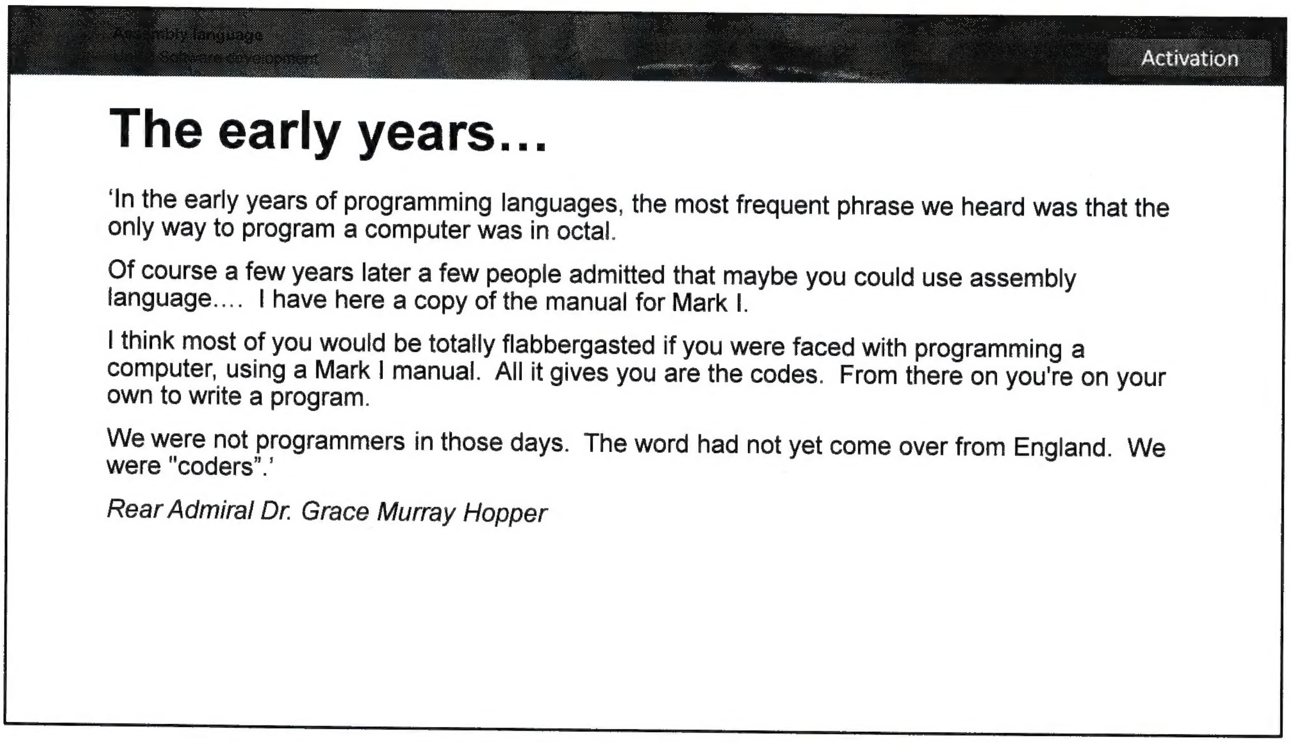
Objectives

- Write and follow simple assembly language programs
- Understand and apply immediate, direct, indirect and indexed addressing modes



Key Words:

- Assembly Language –
  - uses mnemonics to represent the operation codes
- Little Man Computer –
  - has only 11 instructions, and the imaginary computer on which it runs has only 100 memory locations
- Indexed Addressing –
  - the address of the operand is obtained by adding a constant value to the contents of a general register (called the **index register**)



Assembly language
Unit 3 Software development

Activation

The early years…

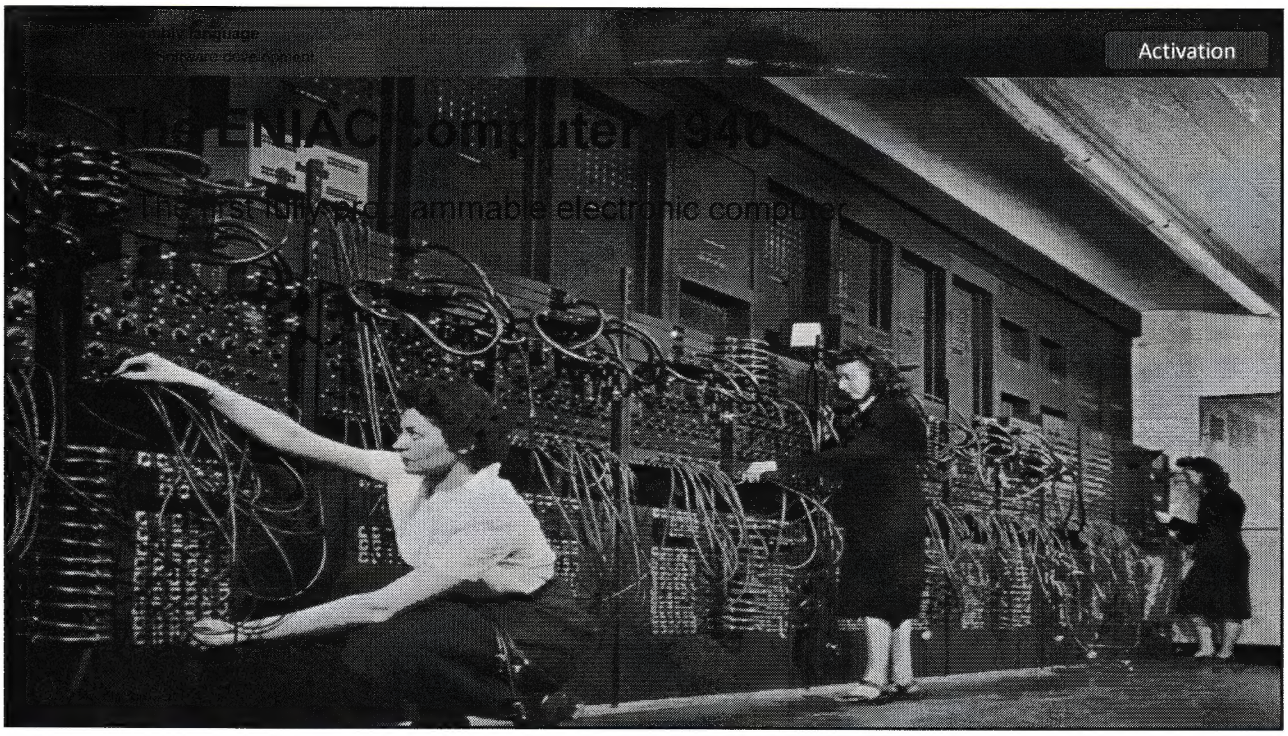
'In the early years of programming languages, the most frequent phrase we heard was that the only way to program a computer was in octal.

Of course a few years later a few people admitted that maybe you could use assembly language.... I have here a copy of the manual for Mark I.

I think most of you would be totally flabbergasted if you were faced with programming a computer, using a Mark I manual. All it gives you are the codes. From there on you're on your own to write a program.

We were not programmers in those days. The word had not yet come over from England. We were "coders".'

Rear Admiral Dr. Grace Murray Hopper



The ENIAC computer 1946

- The first fully programmable electronic computer

Levels of programming languages

- High-level language (Python, VB, etc.):

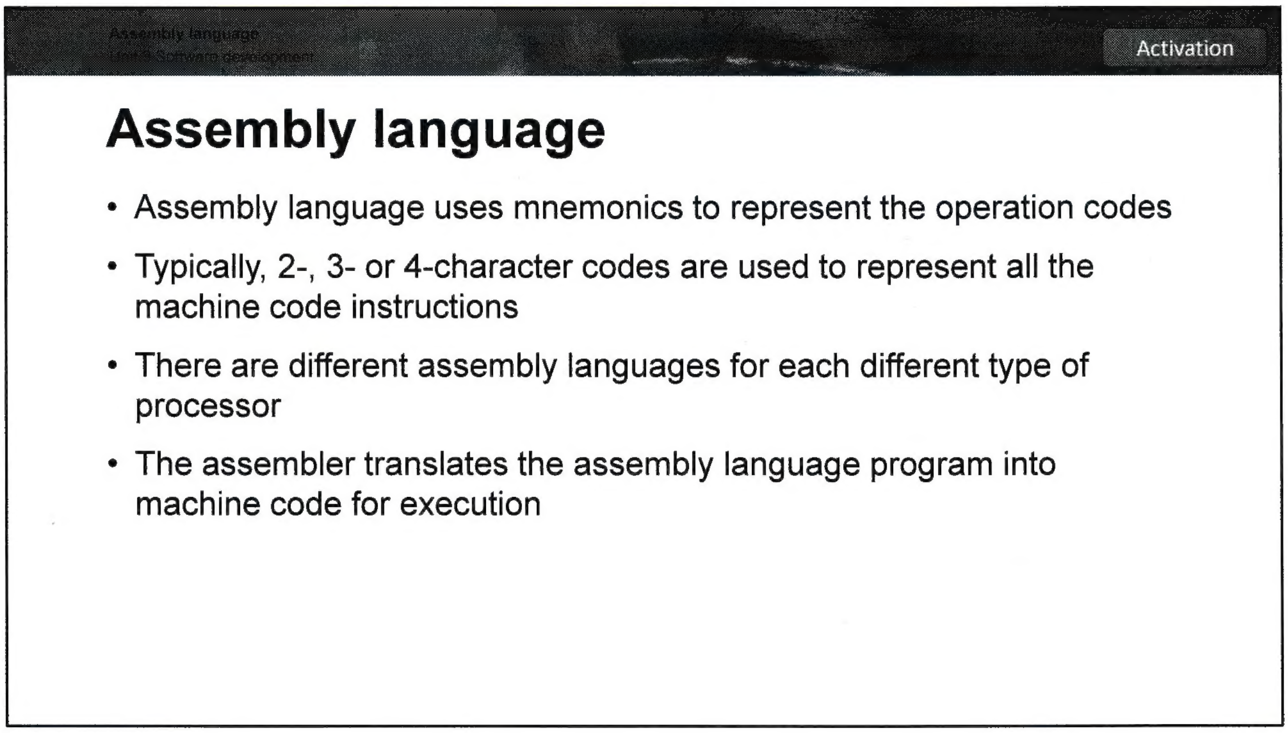
  area = (base * height) / 2

- Low-level language (Assembly language):

  ```
  LDA 501
  ADD 502
  STA  634
  ```

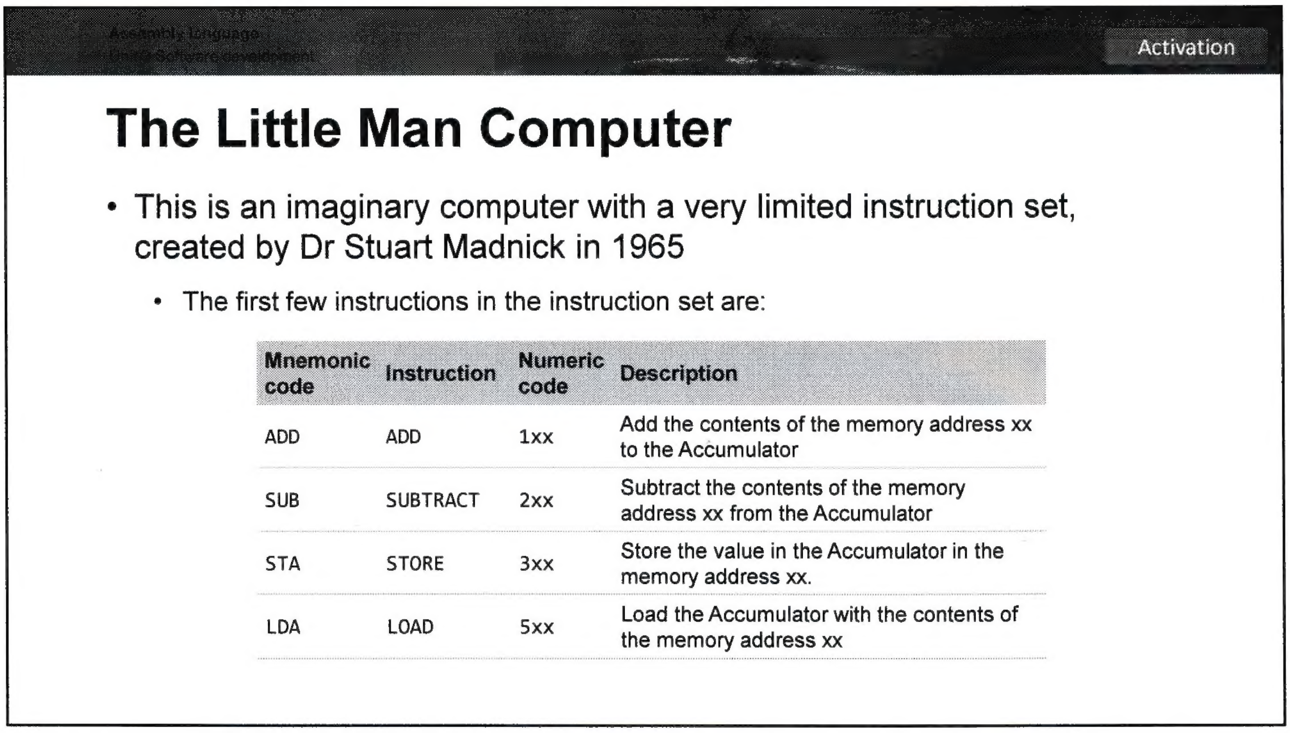
- Machine code:

  ```
  0001011100100011
  0010001011010010
  ```



Assembly language

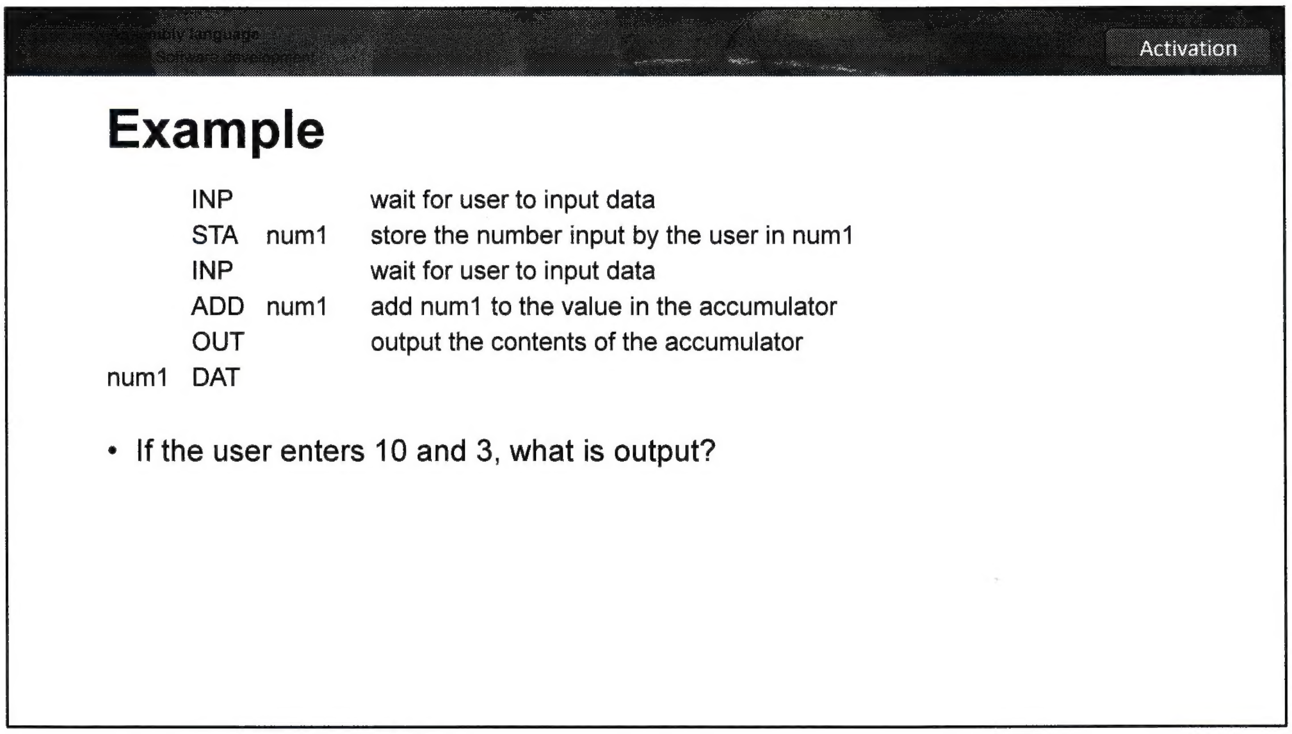
- Assembly language uses mnemonics to represent the operation codes
- Typically, 2-, 3- or 4-character codes are used to represent all the machine code instructions
- There are different assembly languages for each different type of processor
- The assembler translates the assembly language program into machine code for execution



The Little Man Computer

- This is an imaginary computer with a very limited instruction set, created by Dr Stuart Madnick in 1965
  - The first few instructions in the instruction set are:

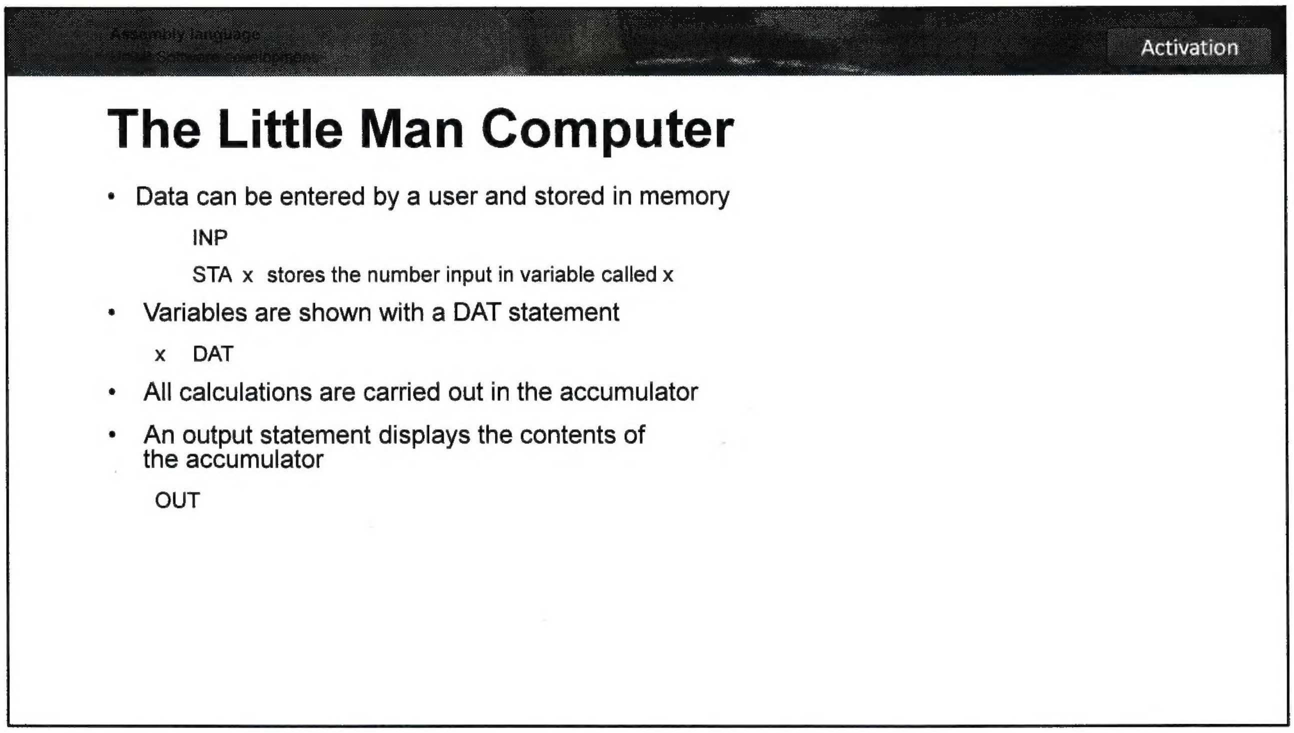
Mnemonic code	Instruction	Numeric code	Description
ADD	ADD	1xx	Add the contents of the memory address xx to the Accumulator
SUB	SUBTRACT	2xx	Subtract the contents of the memory address xx from the Accumulator
STA	STORE	3xx	Store the value in the Accumulator in the memory address xx.
LDA	LOAD	5xx	Load the Accumulator with the contents of the memory address xx



Example

```
        INP            wait for user to input data
        STA   num1     store the number input by the user in num1
        INP            wait for user to input data
        ADD   num1     add num1 to the value in the accumulator
        OUT            output the contents of the accumulator
num1    DAT
```

- If the user enters 10 and 3, what is output?



The Little Man Computer

- Data can be entered by a user and stored in memory

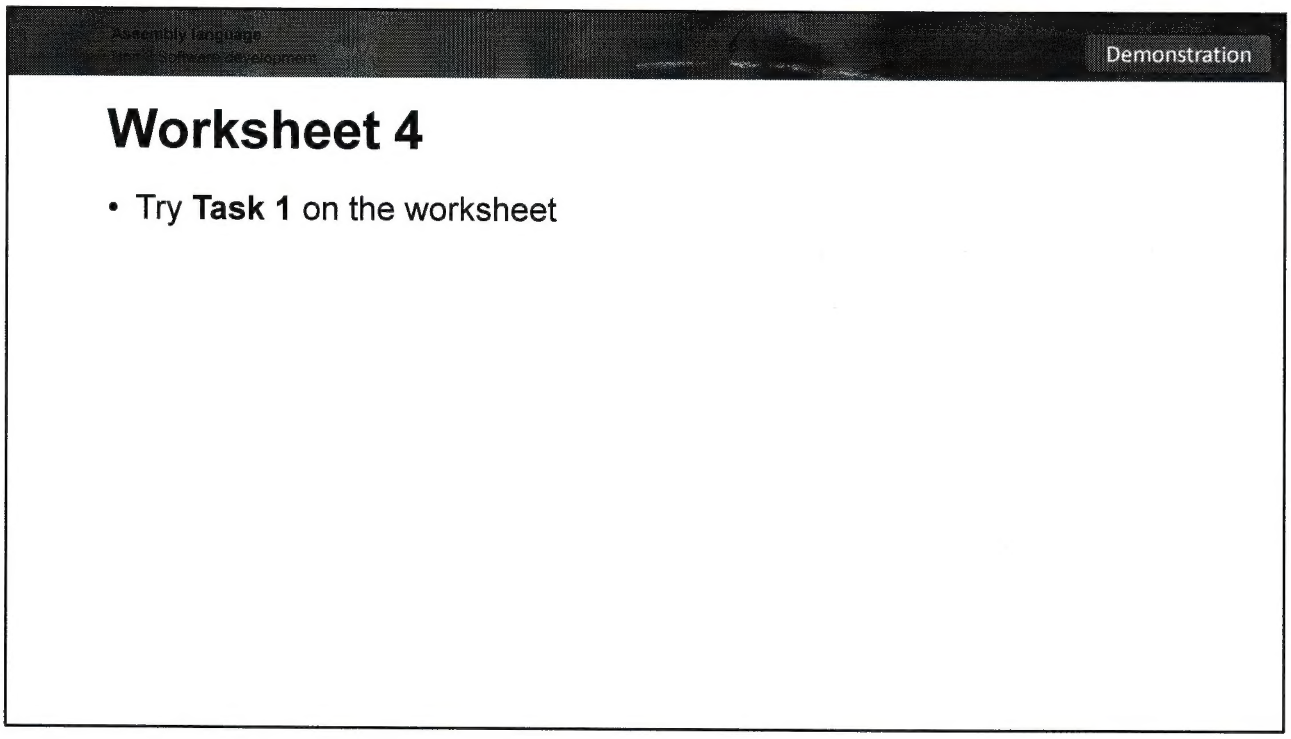
```
INP
STA  x  stores the number input in variable called x
```

- Variables are shown with a DAT statement

```
x    DAT
```

- All calculations are carried out in the accumulator
- An output statement displays the contents of the accumulator

```
OUT
```

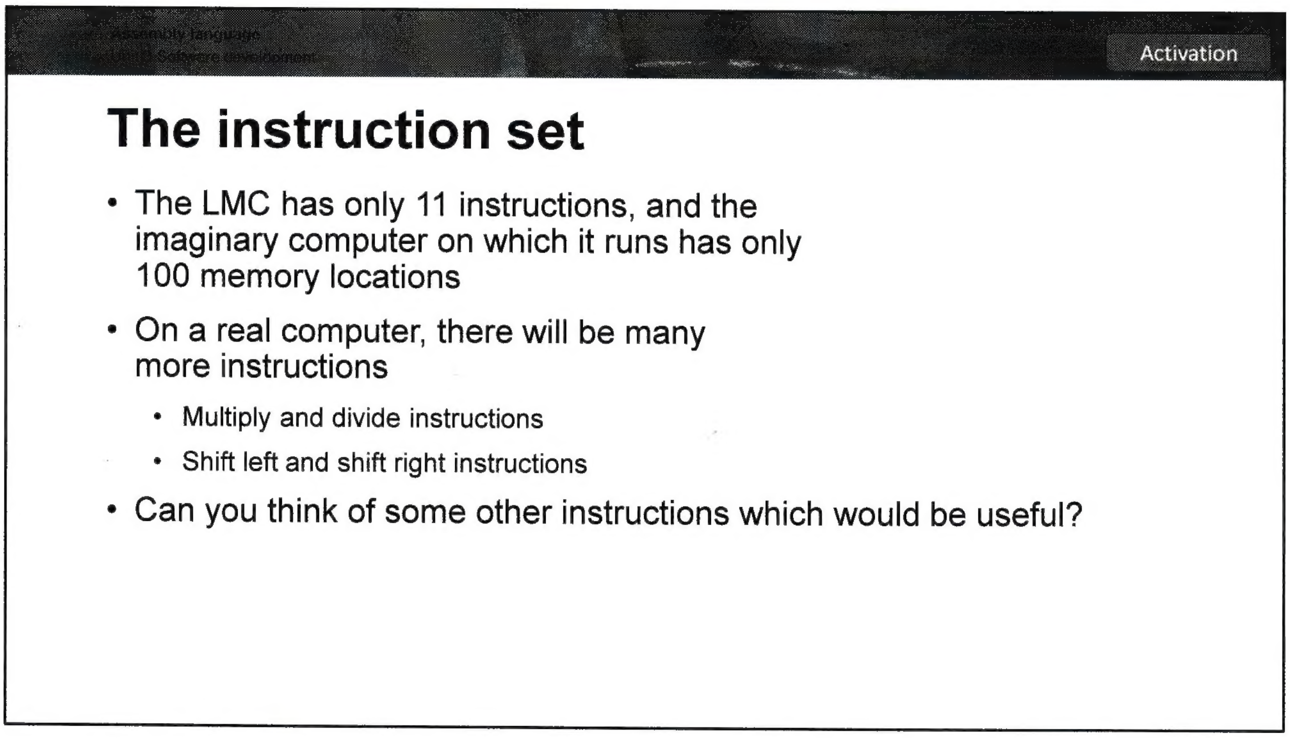


Assembly language
Software development

Demonstration

Worksheet 4

- Try **Task 1** on the worksheet



Assembly language
Software development

Activation

The instruction set

- The LMC has only 11 instructions, and the imaginary computer on which it runs has only 100 memory locations
- On a real computer, there will be many more instructions
  - Multiply and divide instructions
  - Shift left and shift right instructions
- Can you think of some other instructions which would be useful?

Format of a machine code instruction

- A typical machine code instruction, held in 16 bits, could look like this:

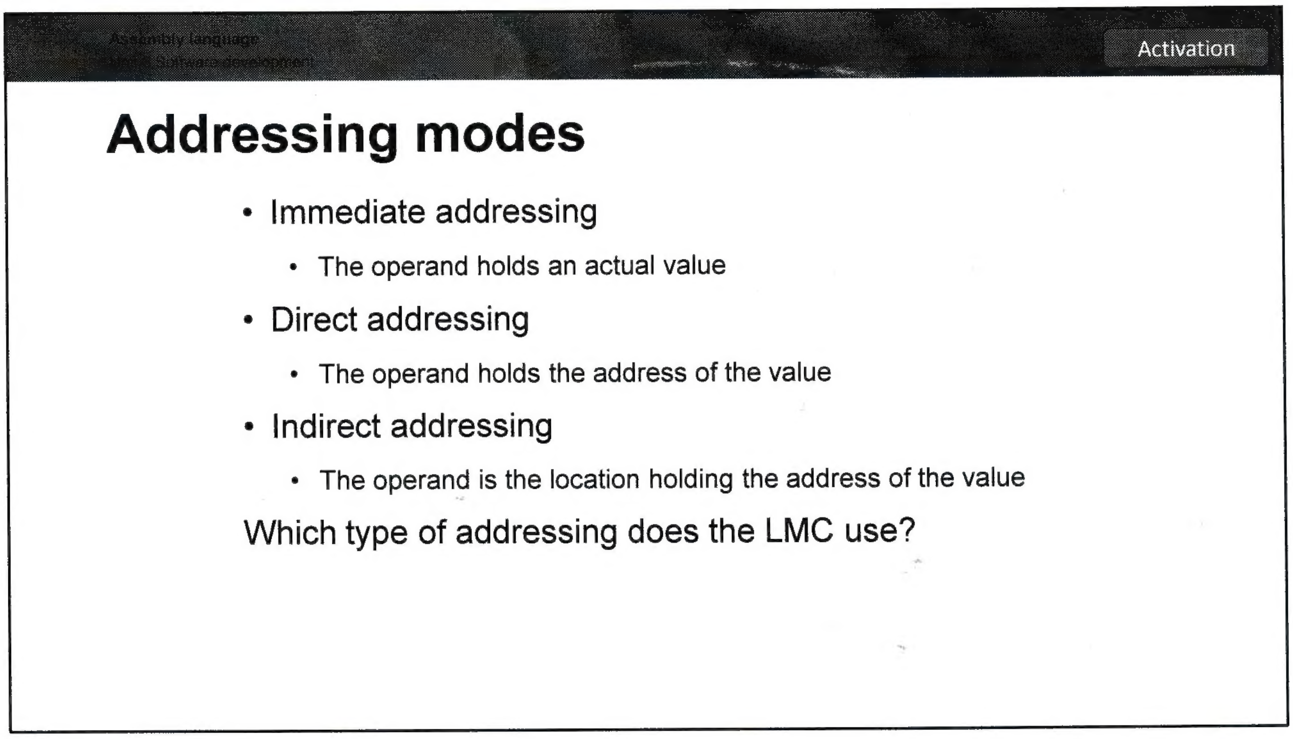
Operation code								Operand(s)							
Basic machine operation						Addressing mode									
0	1	0	0	0	1	0	1	0	0	0	0	0	0	1	1

- How many basic machine operations could this computer have?
- What is the maximum address that could be held as an operand?

Addressing modes

Operation code								Operand(s)							
Basic machine code						Addressing mode									
0	1	0	0	0	1	0	1	0	0	0	0	0	0	1	1

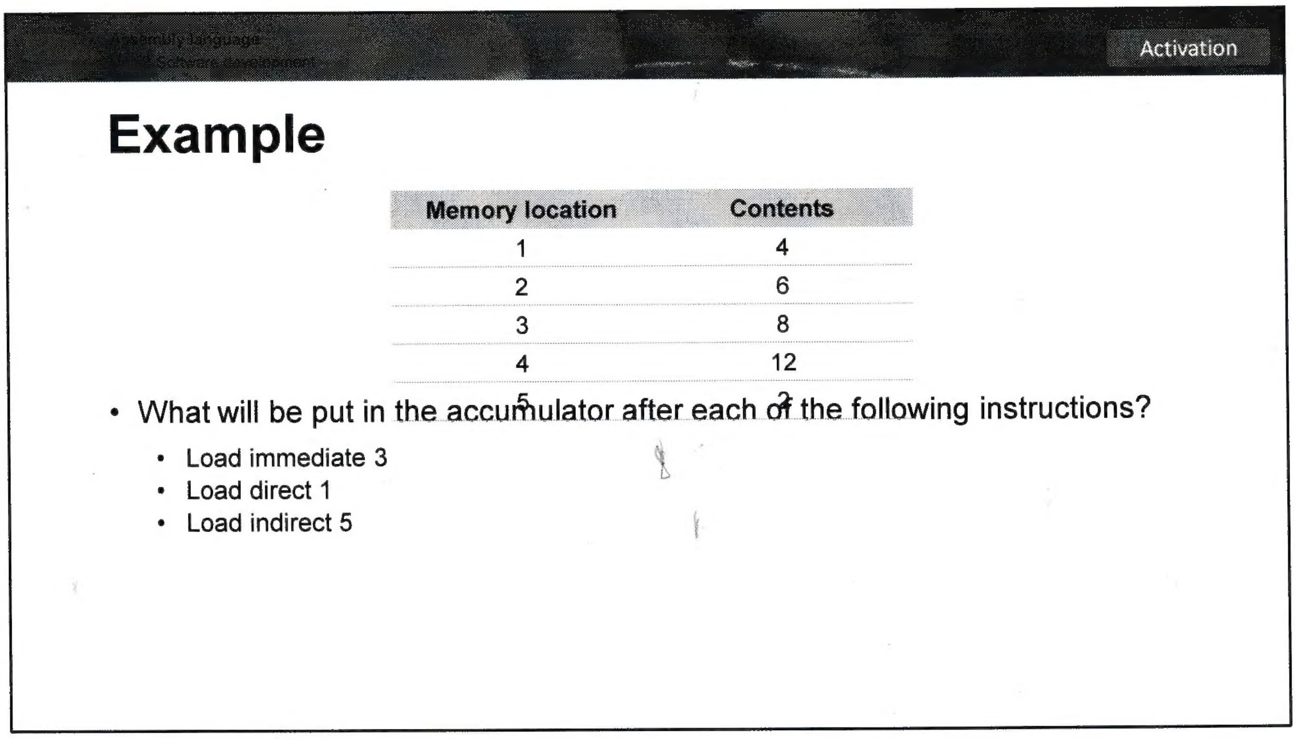
- The last two bits of the operation code ('op code' for short) specify the addressing mode
- This specifies whether the operand represents
  - An actual value to be used in a calculation
  - The memory address of a value to be used
  - The address of a register or memory location which holds the memory address of the value to be used, or
  - An index (see later slide)



Addressing modes

- Immediate addressing
  - The operand holds an actual value
- Direct addressing
  - The operand holds the address of the value
- Indirect addressing
  - The operand is the location holding the address of the value

Which type of addressing does the LMC use?



Example

Memory location	Contents
1	4
2	6
3	8
4	12
5	2

- What will be put in the accumulator after each of the following instructions?
  - Load immediate 3
  - Load direct 1
  - Load indirect 5

Activation

Indexed addressing

- Using indexed addressing, the address of the operand is obtained by adding a constant value to the contents of a general register (called the **index register**)
- Indexed addressing mode is used to access an array whose elements are in successive memory locations
- By incrementing the value in the index register, successive memory locations can be accessed

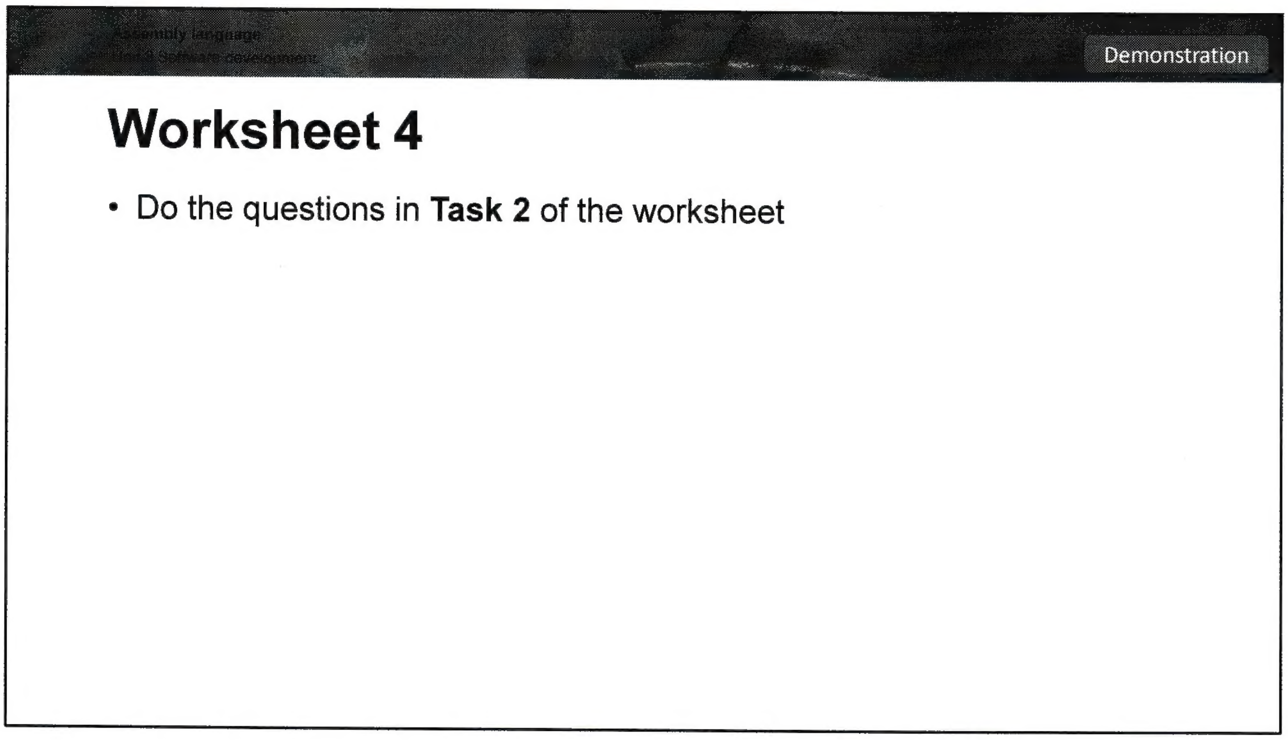
Activation

Example

- Assume register R0 contains 0 and the index register contains 31

Memory location	Contents
31	4
32	6
33	8
34	12
35	2

- **Load indexed R0** will put the contents of location 31 (i.e. contents of R0 + index register) into accumulator
- Increment **R0**
- **Load indexed R0** will now put the contents of location 32 (i.e. contents of R0 + index register) into accumulator

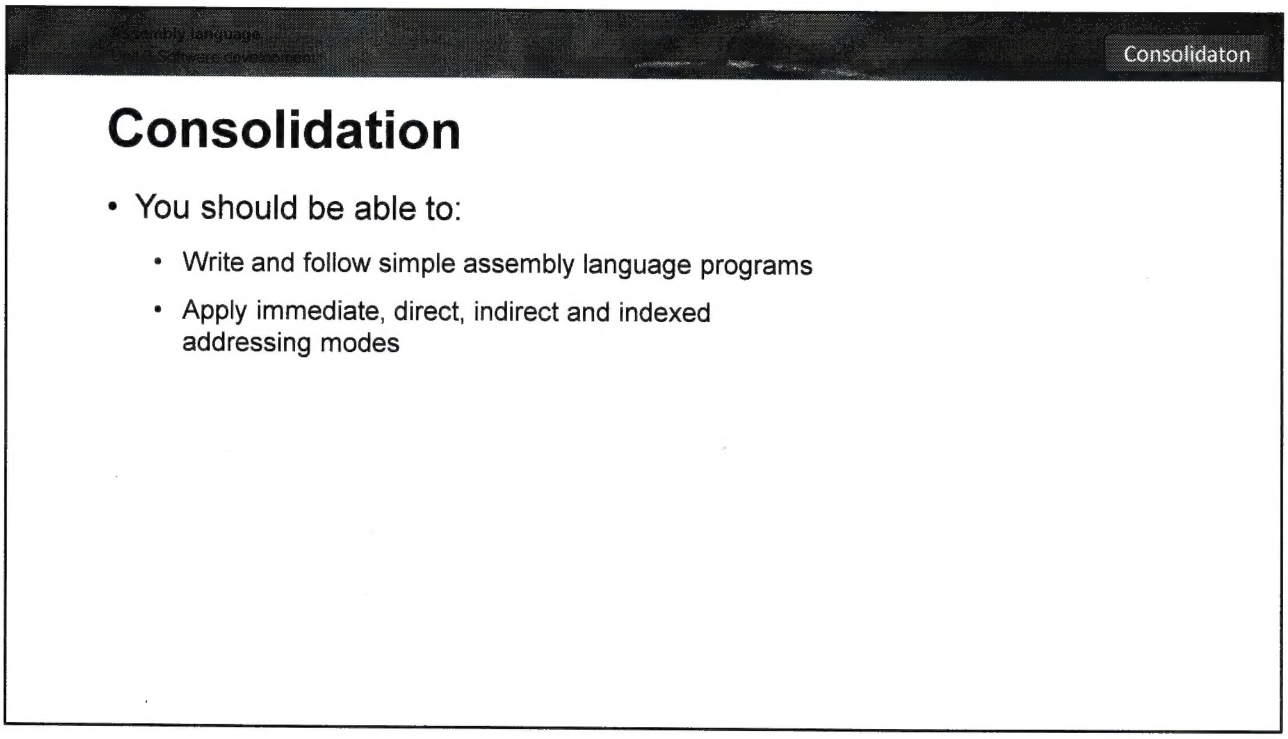


Assembly language
Unit 2 Software development

Demonstration

Worksheet 4

- Do the questions in **Task 2** of the worksheet



Assembly language
Unit 2 Software development

Consolidaton

Consolidation

- You should be able to:
  - Write and follow simple assembly language programs
  - Apply immediate, direct, indirect and indexed addressing modes